Scene Text Recognition for Augmented Reality

Sagar G V
Adviser: Prof. Bharadwaj Amrutur
Indian Institute Of Science

Outline

Research area and motivation

- Augmented reality now viable thanks to advances in display technology and compute power
- Computer vision now actually works
- What APIs could developers use to build rich AR applications?
- Desktop : Browser :: AR : ?

Finding text in natural scenes

- Lots of objects are labelled with text
- Useful API for Augmented Reality devs
- Number plate, ID card scanning
- Reduces need for QR codes

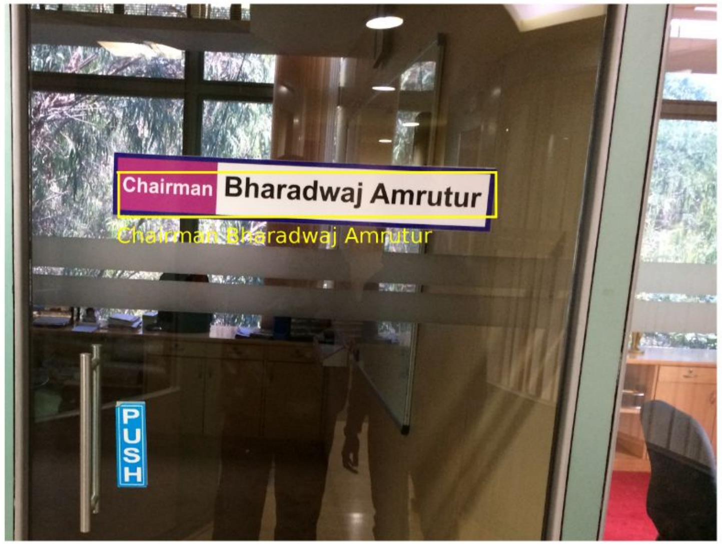


Figure 1: Recognizing text in a nameplate.

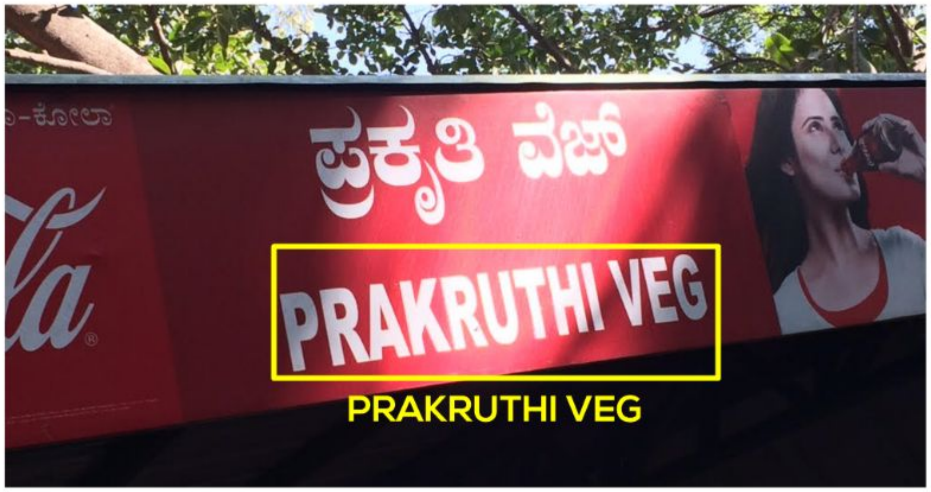


Figure 2: Recognizing text in a billboard.

Prior art

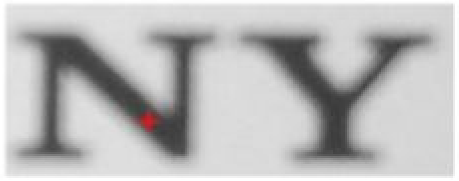
- Automatic Number Plate Recognition (ANPR)
    - OpenALPR: http://www.openalpr.com/
- Extremal Region (ER) based character detector
    - L. Neumann, J. Matas, "Real-time Lexicon-free Scene Text Localization and Recognition", TPAMI 2015.
- Bottom-up convolutional network to detect characters
    - M. Jaderberg, A. Vedaldi, A. Zisserman, “Deep Features for Text Spotting”, ECCV 2014.
- End-to-end convolutional neural network to localize text
    - A. Gupta, A. Vedaldi, A. Zisserman, “Synthetic Data for Text Localisation in Natural Images”, CVPR 2016.
- Recurrent neural network to localize text
    - Z. Tian et. al, “Detecting text in natural image with connectionist text proposal network”, ECCV 2016.

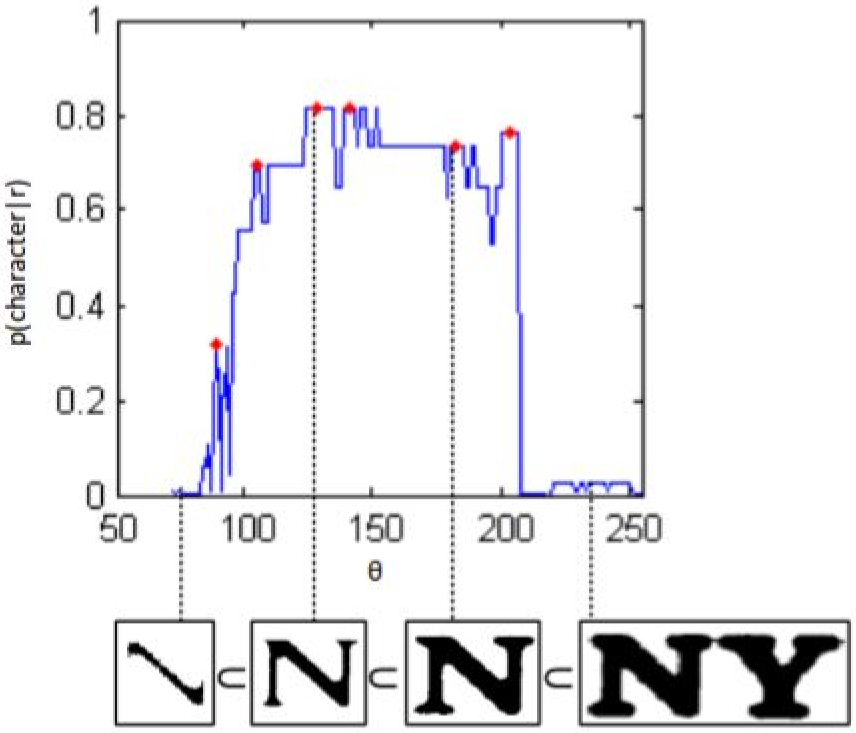
Automatic Number Plate Recognition

- Region proposal
- Binarization
- Character analysis for early rejection
- Edge detection
- Deskewing
- Character segmentation
- OCR
- Post processing

ER based text detection

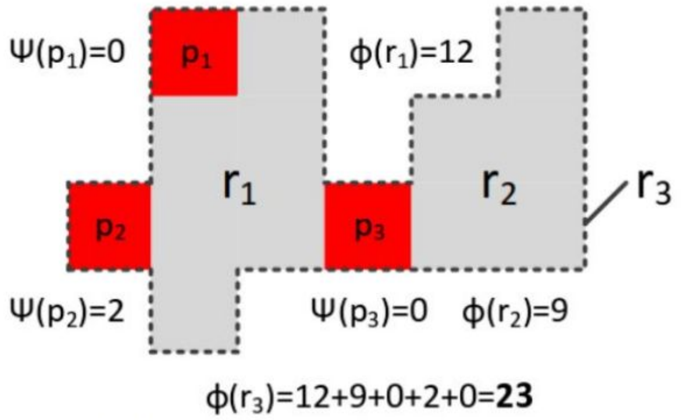
- Explore all connected regions at all thresholds efficiently using union-find
- Text/no-text classifier (cascade) looks at region



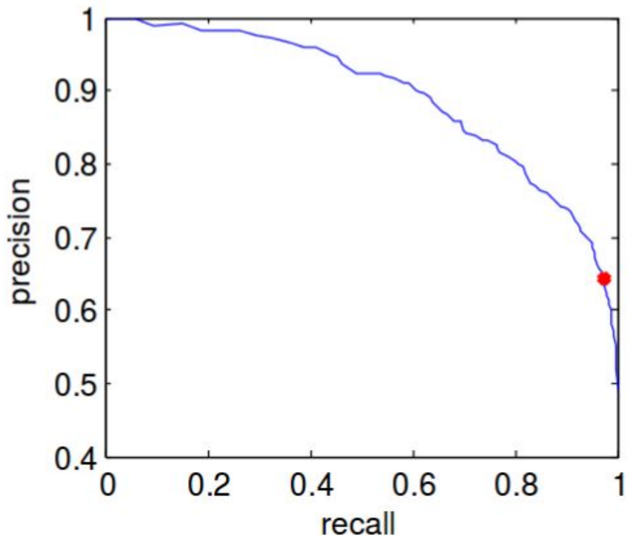


ER based text detection

- First stage of text/no-text classifier cascade uses only incrementally computable features: area, perimeter, aspect ratio, euler number.
- Tuned for high recall
- Second stage uses an SVM

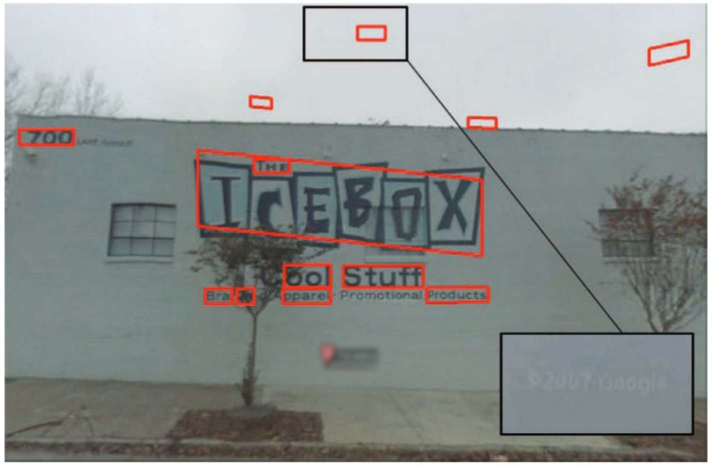


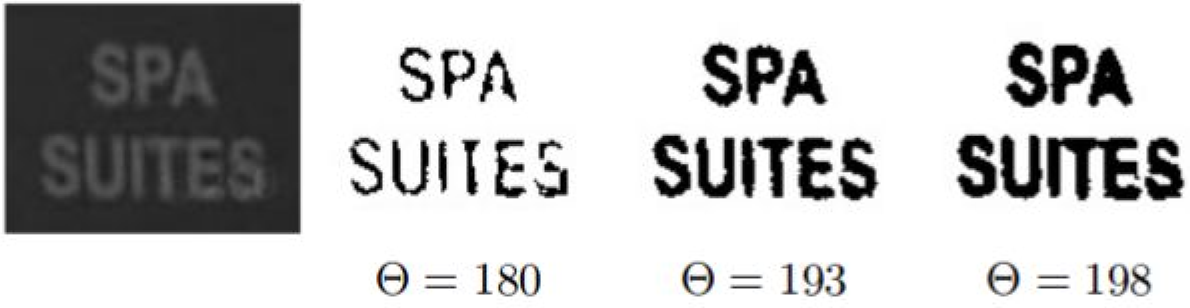
(a) perimeter p



ER based text detection

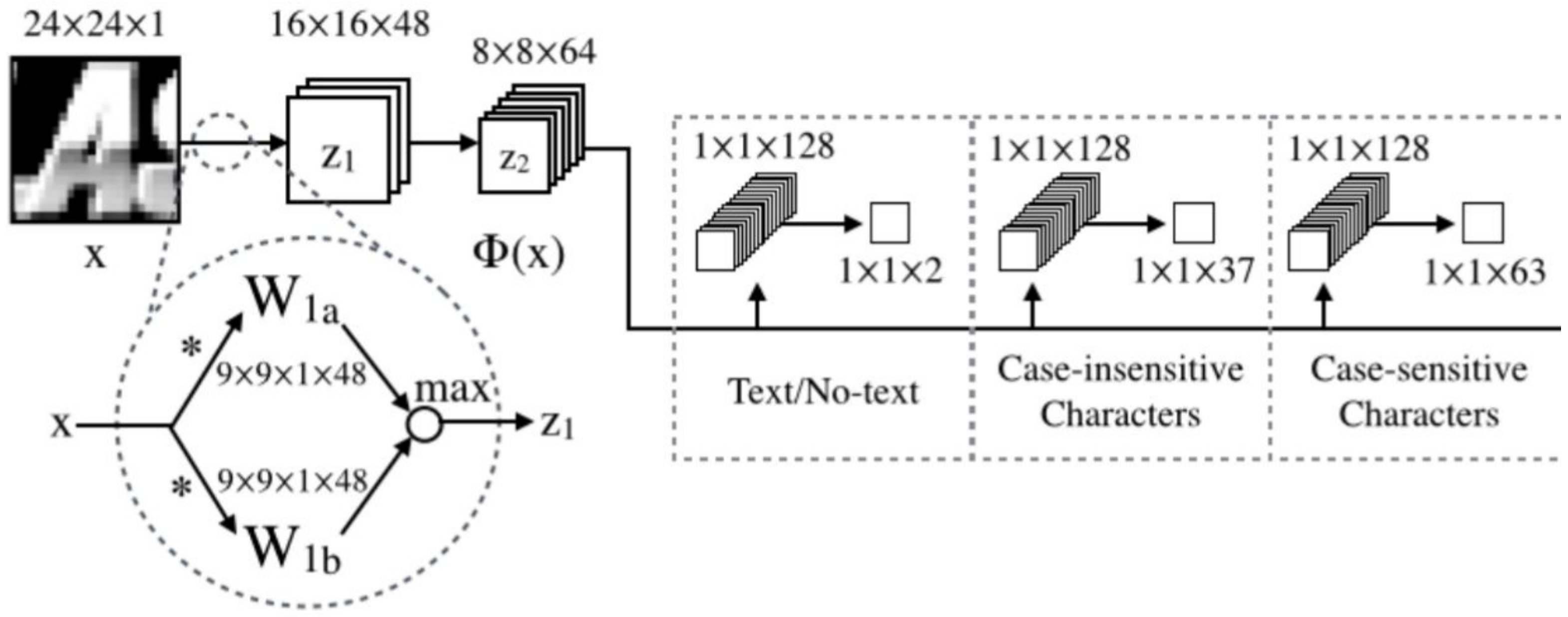
- Detector is scale invariant
- ERs provide segmentation as well
- Can find tiny low contrast watermarks too!
- Difficult to accelerate on GPUs
- Sensitive to blur, occlusion





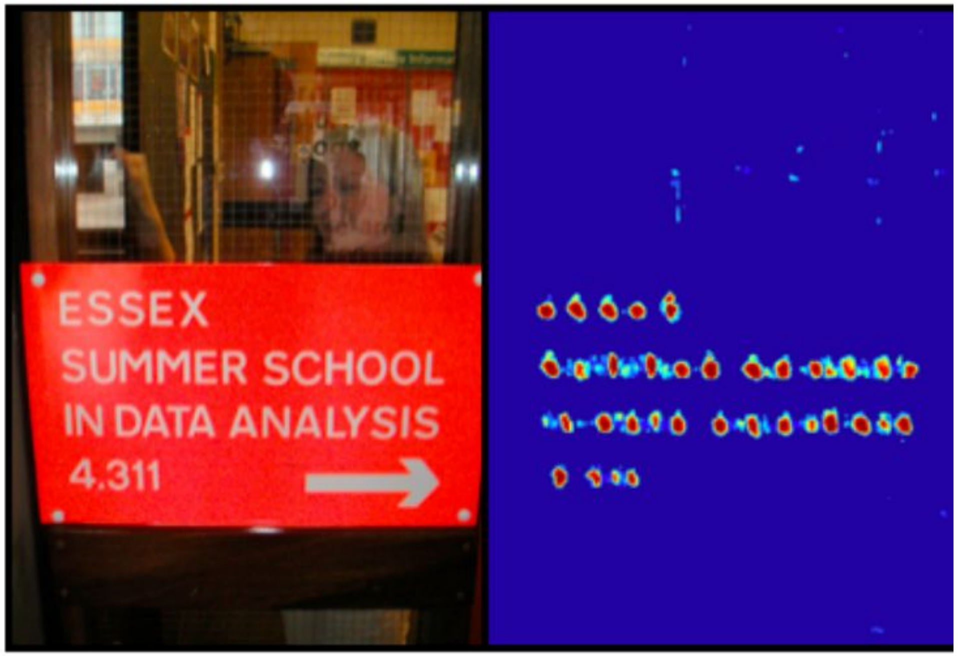
Bottom-up CNN based text detector

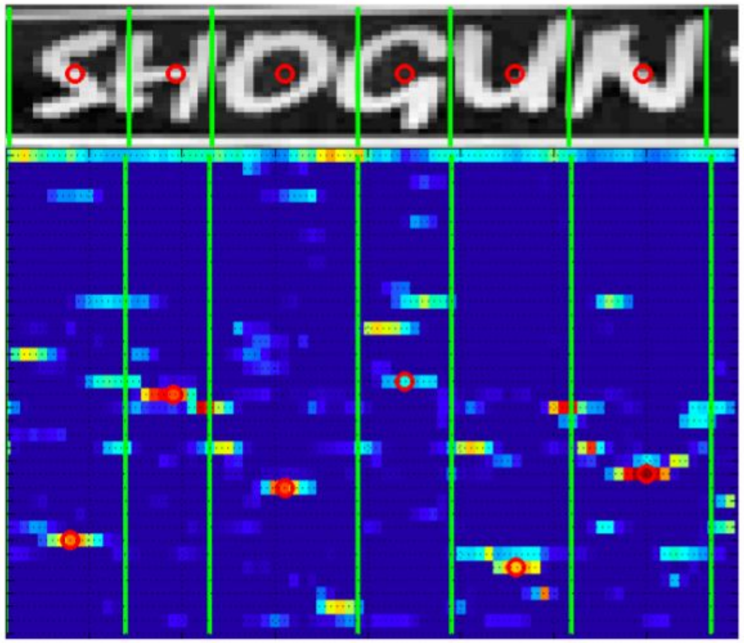
- Slide over 24x24 windows
- Convolutional layers share compute with adjacent windows
- Attach 3 different heads after conv layers



Bottom-up CNN based text detector

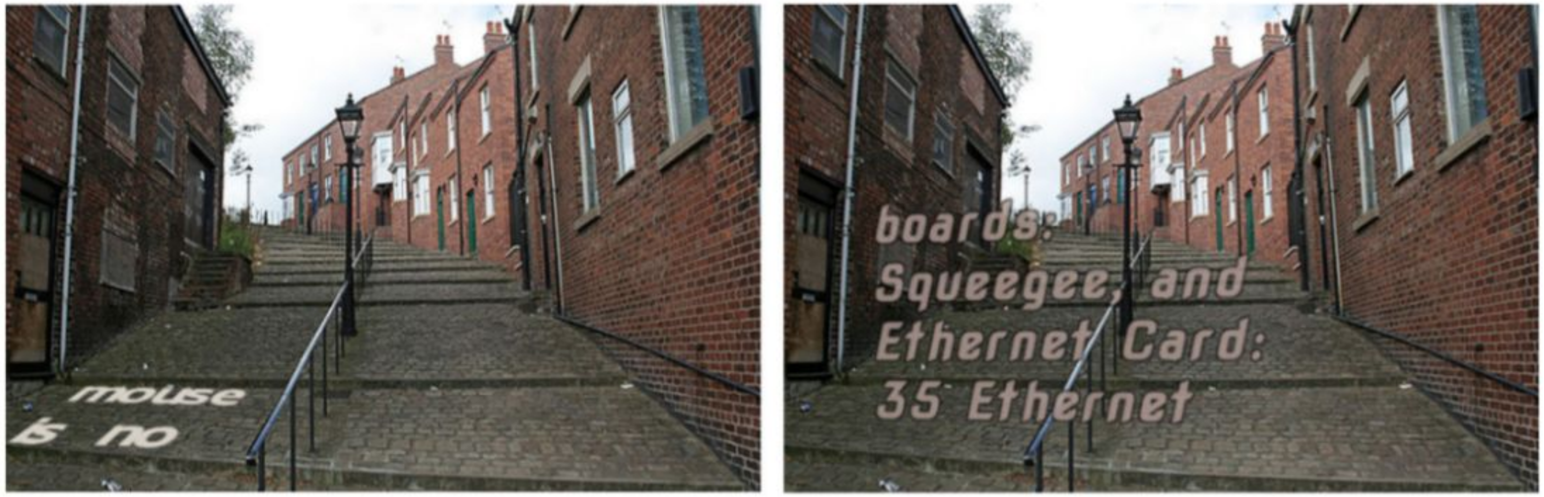
- The detector is fast
- Easy to accelerate on GPU
- High false positive rate





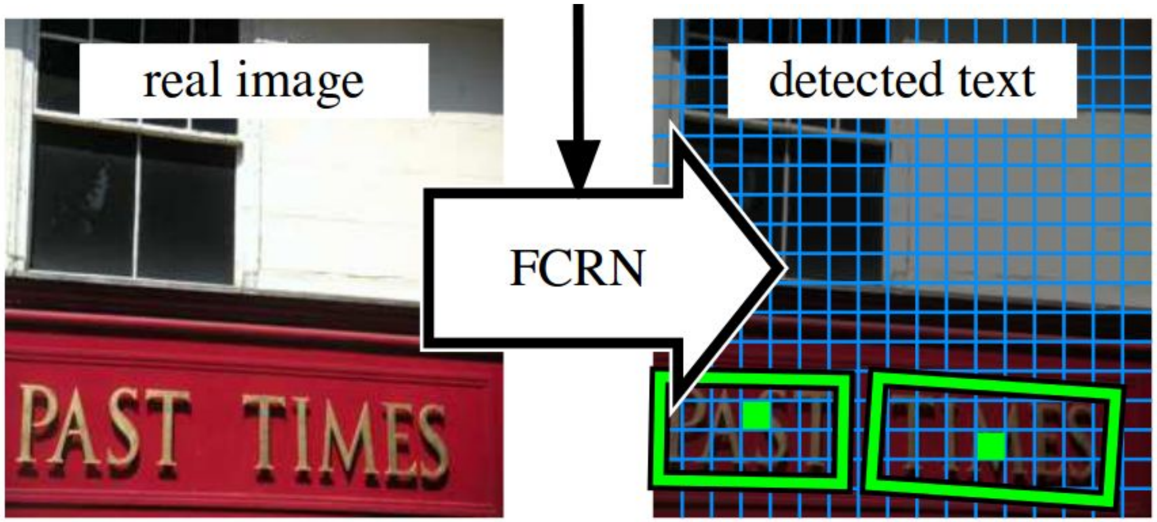
End-to-end CNN for text localization

- Generate training dataset from natural images
- Composite text with background after affine transformation and adding shadow
- Segment the image and have text respect segment boundaries



End-to-end CNN for text localization

- Deep CNN to directly predict word bounding boxes
- CR64-5x5, MP2, CR128-5x5, MP2, CR128-3x3, CR128-3x3, MP2, CR-256-3x3, CR-256-3x3, MP2, CR-512-3x3, CR-512-3x3, CR-512-5x5, tensor 7x1 - object presence and $\mathbf{p} = (x-u, y-v, w, h, \cos\theta, \sin\theta)$
- Network is fully convolutional (i.e. no FC layer that looks at the whole image for each prediction)
- Moderately fast (70 ms per image on a desktop GPU).



Recurrent neural network for text localization

- Generic object detector has trouble getting word boundaries right
- They don't take advantage of the fact that text usually appears in lines
- Look for text lines and fine vertical "text pieces"

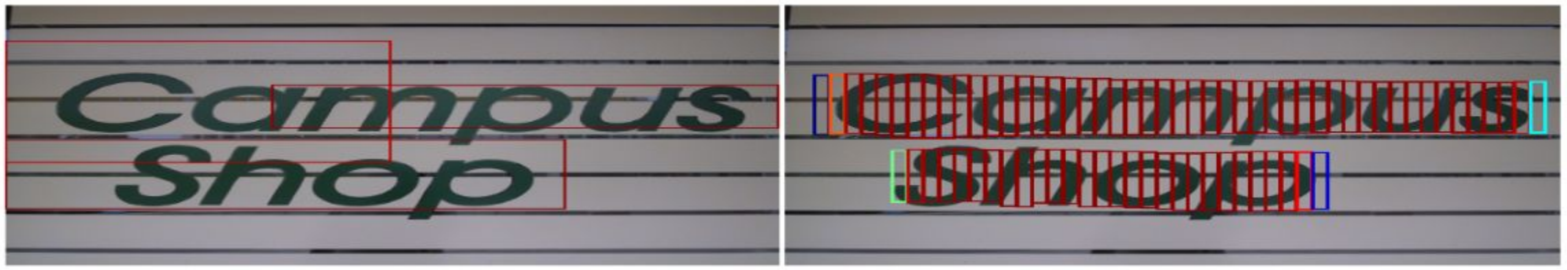
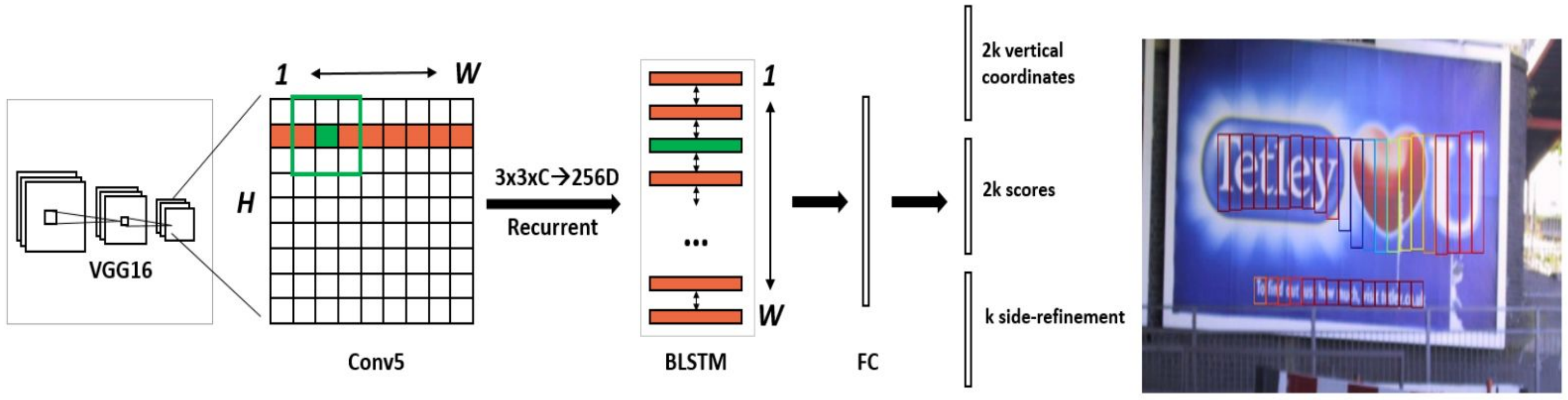


Fig. 2: **Left**: RPN proposals. **Right**: Fine-scale text proposals.

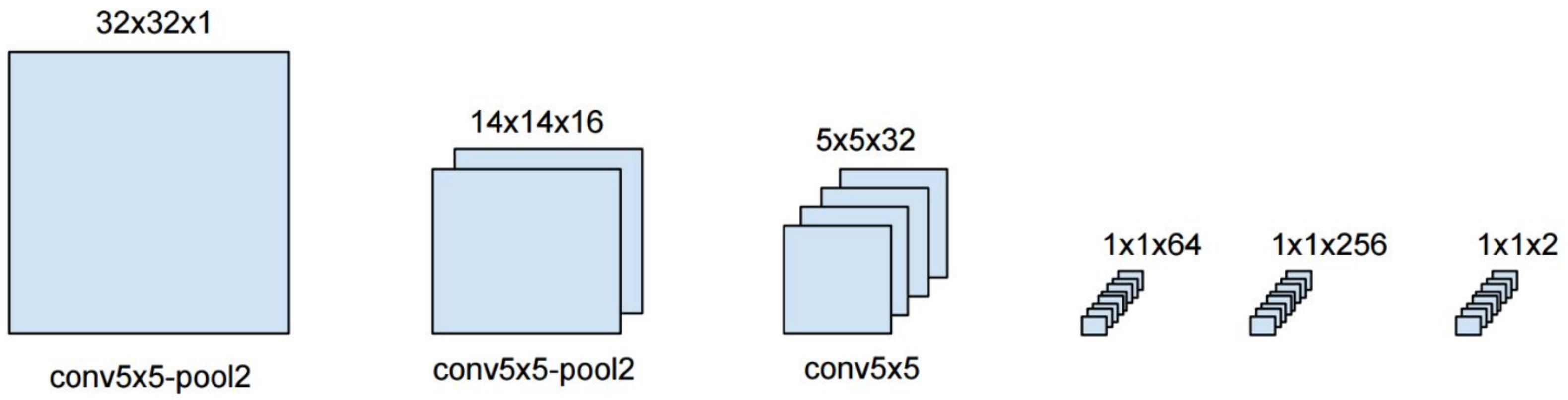
Recurrent neural network for text localization

- Densely slide 3x3 window over VGG-16 conv map
- Feed the row to a BLSTM
- Predict k text/no-text scores, y-axis coordinates, and side refinements



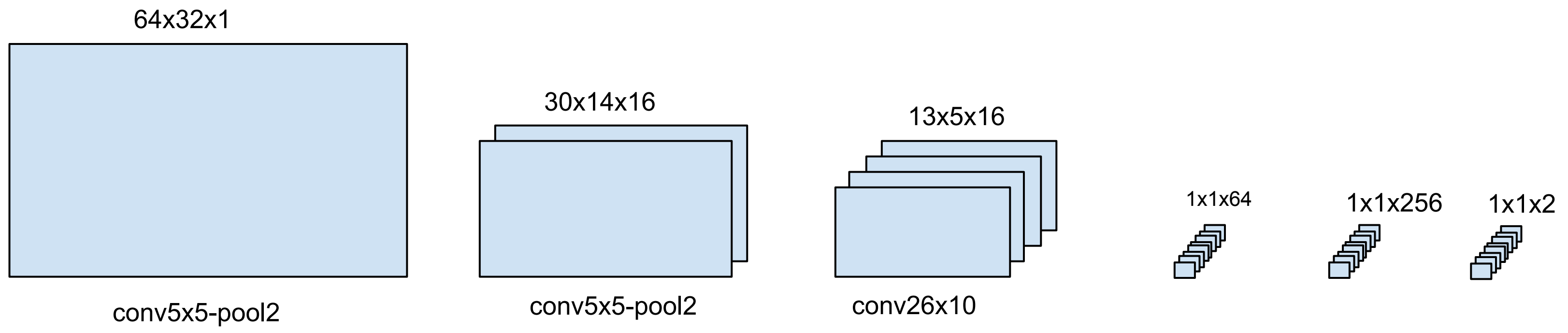
Recap: Sliding window detector

- CNN based sliding window detector for characters is fast
- But false positive rate is high because there is too little context
- Lots of background patches look like characters!



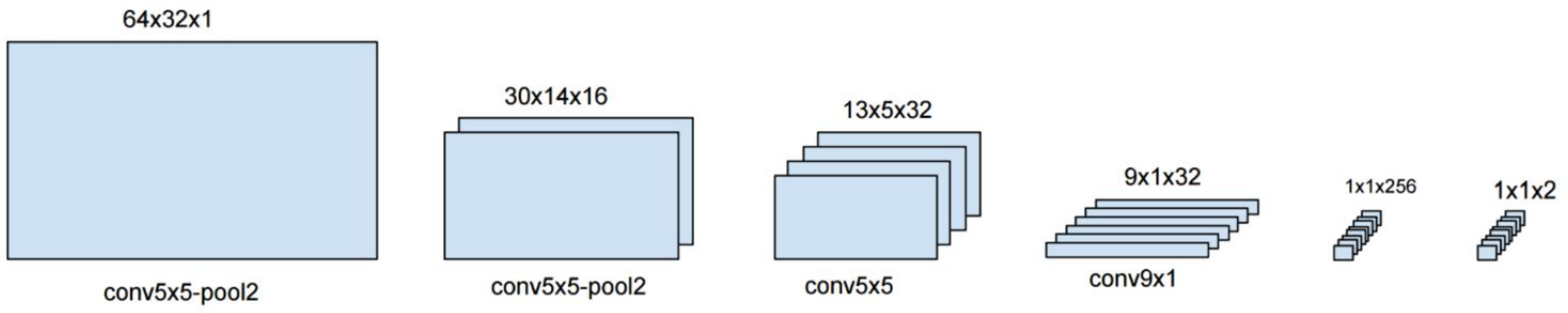
Improving the sliding window detector

- Proposal: Look for bigrams (character pairs) rather than single characters
- Naive network has twice the number of inputs at the first FC layer



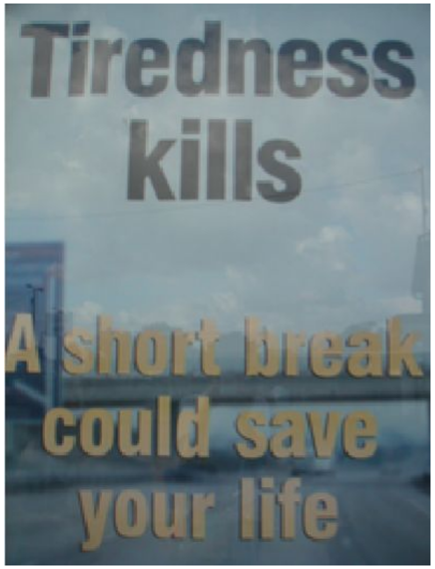
Improving the sliding window detector

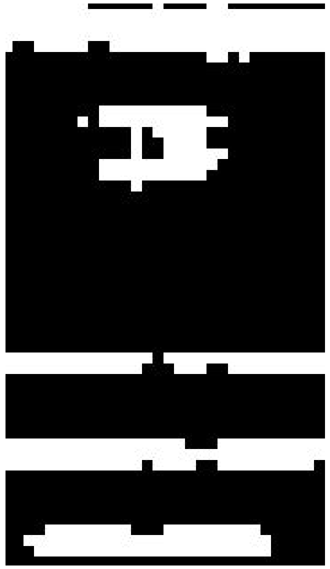
- Proposal: Look for bigrams (character pairs) rather than single characters
- Have a layer that looks at the output of conv map having character features
- Shares more computation between adjacent windows

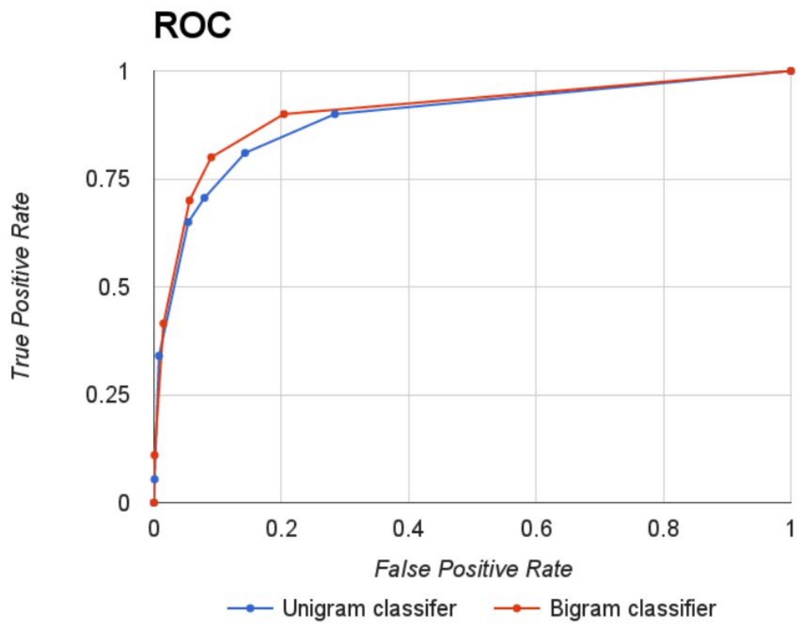


Improving the sliding window detector

- Bigram detector has ~25% more MAC ops/px than character detector
- False positive rate reduced from 28.4% to 20.4% at 90% precision on the ICDAR 2015 dataset

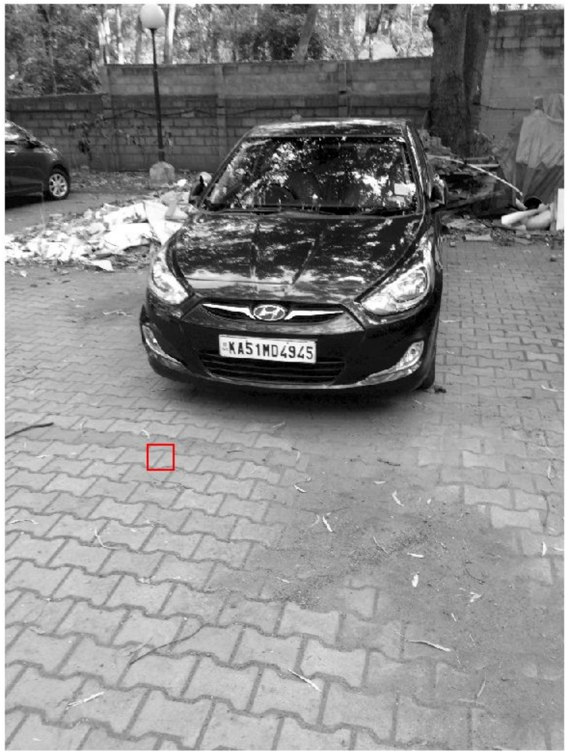


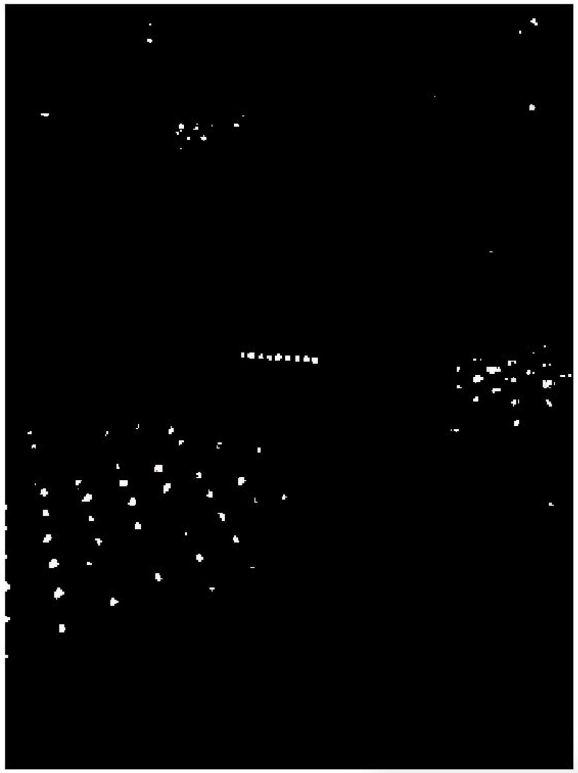




Number plate recognition using character detector

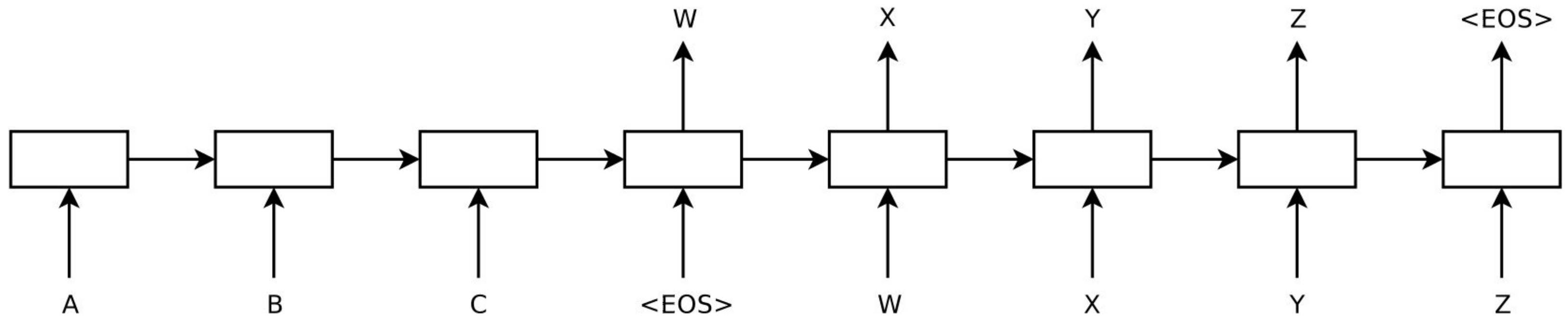
- Do false positives matter so much if a language model is available?
- A regex match can eliminate all false positive number plates in most images!





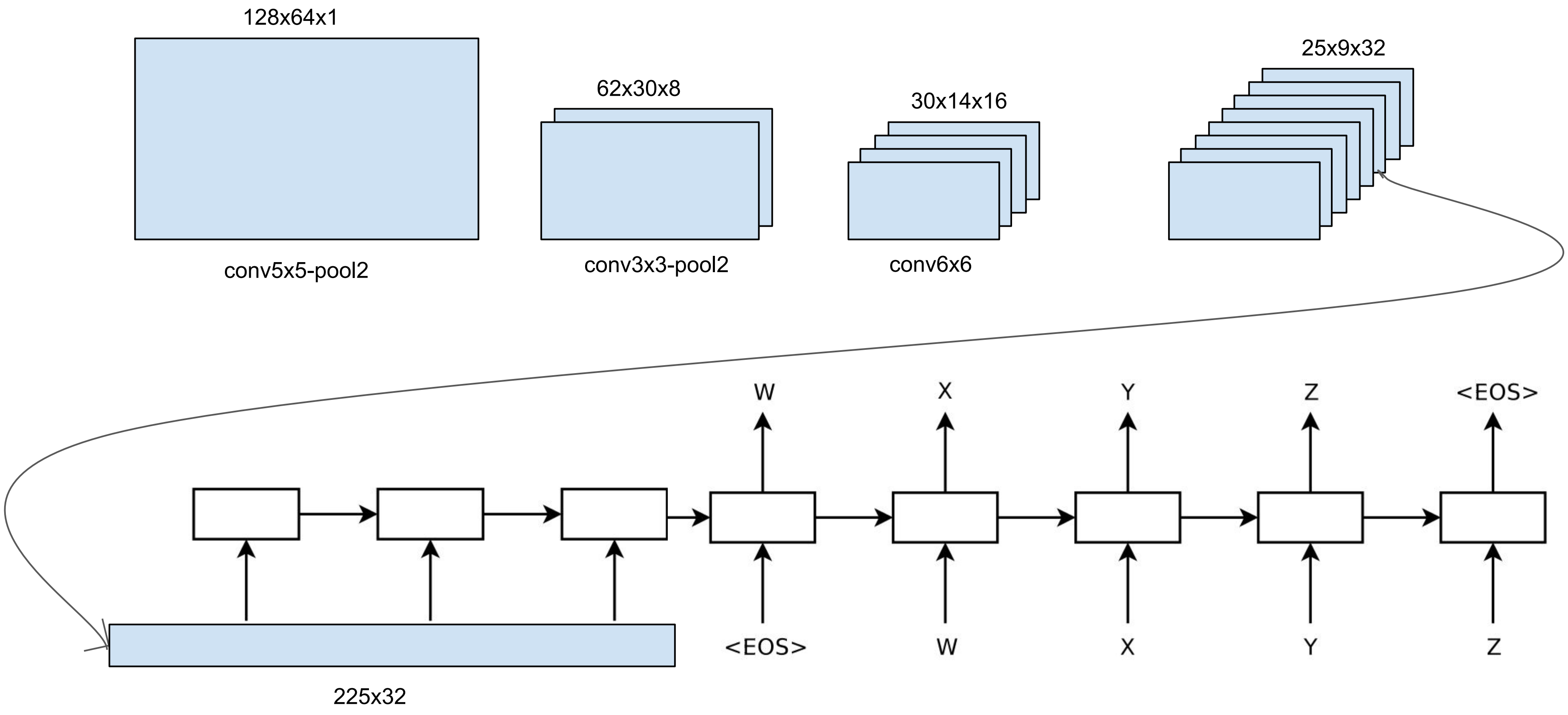
Encoder-decoder network

- Encoder-decoder recurrent networks have performed well for machine translation
- The entire source sentence is captured in the state of a recurrent network which then generates the translated sequence



End-to-end text recognition

- Encode an entire image into a vector that contains all the information about text in the image.



Future work

- Explore possibility of end-to-end text detection
- Simultaneous detection and localization
- Incorporate language model (constructed from non-image text corpus)
- Extend detection to videos

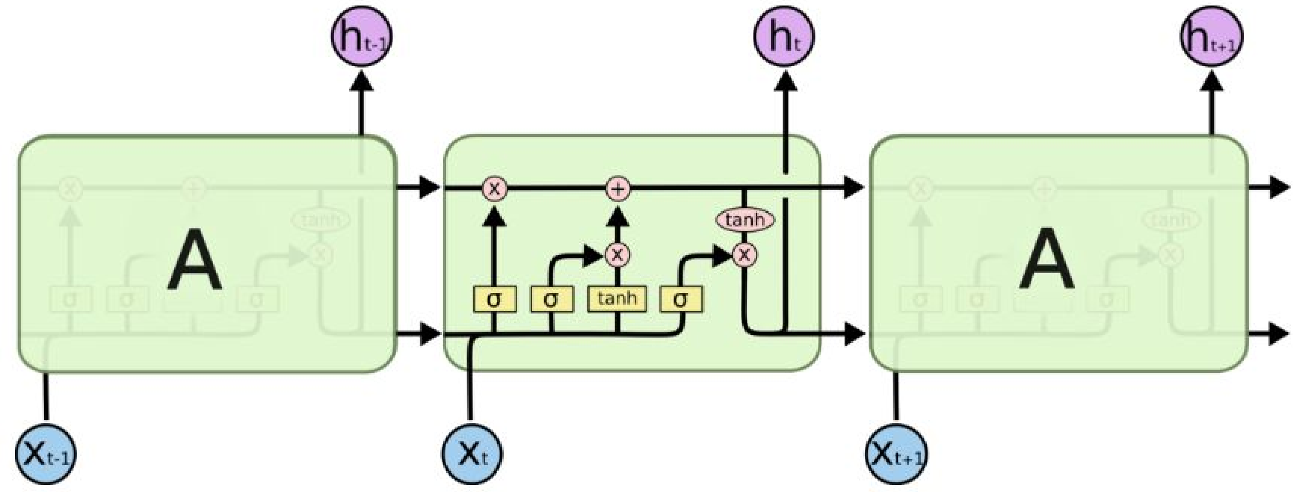
Acknowledgements

- Thanks to NVidia for providing the GPU used in this work

Conclusion

- Finding text in natural scenes is a useful tool to build AR apps
- Performance of sliding window detector can be improved by looking for bigrams rather than individual characters
- Careful choice of network architecture can reduce the overhead of detecting characters pairs
- End-to-end detection has the benefit of more context and can perform better than a multi-stage pipeline with separate cost per stage

Extra slides - LSTM



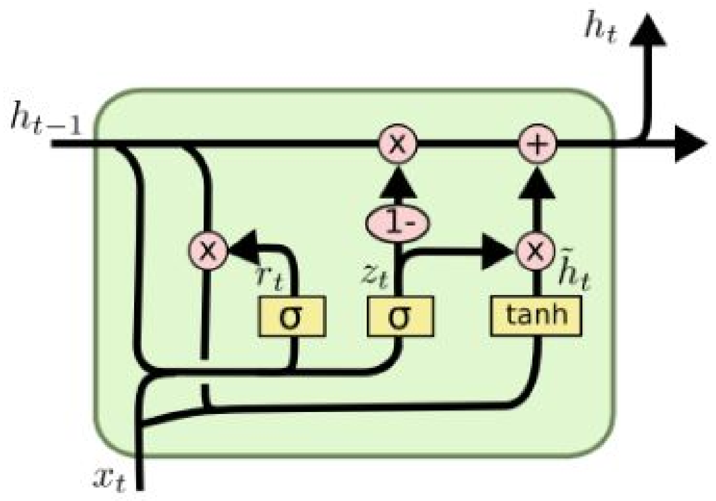
$$i_t = \sigma\left(W^{(i)}x_t + U^{(i)}h_{t-1}\right)$$
$$f_t = \sigma\left(W^{(f)}x_t + U^{(f)}h_{t-1}\right)$$
$$o_t = \sigma\left(W^{(o)}x_t + U^{(o)}h_{t-1}\right)$$
$$\tilde{c}_t = \tanh\left(W^{(c)}x_t + U^{(c)}h_{t-1}\right)$$

$$c_t = f_t \circ c_{t-1} + i_t \circ \tilde{c}_t$$

$$h_t = o_t \circ \tanh(c_t)$$

Illustration: http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Extra slides - GRU



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

Illustration: http://colah.github.io/posts/2015-08-Understanding-LSTMs/